ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(НИУ «БелГУ»)

ИНСТИУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА ОБЩЕЙ МАТЕМАТИКИ

ОНЛАЙН КАЛЬКУЛЯТОР ДЛЯ ОПРЕДЕЛЕНИЯ РЕЙТИНГОВ СЛОЖНЫХ ОБЪЕКТОВ ПО КОМПЛЕКСУ ПОКАЗАТЕЛЕЙ

Выпускная квалификационная работа
обучающегося по направлению подготовки
01.03.02 Прикладная математика и информатика
очной формы обучения, группы 07001406
Мандрикова Дмитрия Александровича

Научный руководитель
д.т.н., профессор
Аверин Г.В.

БЕЛГОРОД 2018

ВВЕДЕНИЕ

Информация — результат и отражение в человеческом сознании, многообразие внутреннего и окружающего миров (сведения об окружающих человека предметах, явлений, действия других людей). Обладание достоверной информацией играет важнейшую роль в организации процесса управления.

Потребность в передаче и хранении различного рода информации сформировалась и развивалась вместе с эволюцией человеческого общества. Сегодня можно смело утверждать, что сфера деятельности человека в области информационных и телекоммуникационных технологий, является определяющим фактором интеллектуальной, экономической и оборонной возможностей государства и человеческого общества в целом.

В настоящее время более 50% населения Земли имеет доступ к интернету. Это дает возможность быстро обмениваться информацией на больших расстояниях.

Современные технологии дают возможность использовать многие программные продукты в режиме онлайн, что является очень удобным способом производить нужные операции в любом месте, в котором имеется доступ в интернет.

Целью данной ВКР является разработка онлайн калькулятора для определения рейтингов сложных объектов по комплексу показателей.

Для достижения поставленной цели в работе решены следующие задачи:

- изучено применение математических методов в термодинамике, проанализированы численные методы и программные продукты для вычислений термодинамических величин;
- рассмотрены методы дифференциальной геометрии в приложении к термодинамике, геометрические аксиомы, положения и вычислительные задачи термодинамики;

- выбран программный продукт для реализации вычислительных экспериментов и разработан алгоритм на основе средств дифференциальной геометрии;
- сформированы задачи перспективных исследований.

Для решения поставленных задач использовались следующие методы:

- систематизация источников информации по теме исследования;
- метод системного анализа применительно к описанию процессов;
- метод дифференциальной геометрии, функционального анализа, методика решений дифференциальных уравнений;
- математическое моделирование и расчетные компьютерные методы применительно к исследуемым процессам;
- методы обобщения информации и статистической обработки данных;
- апробация методов.

Решение задач многокритериального выбора является сложным, трудоемким и ресурсоемким процессом, поэтому разработка инструментов для автоматизации их решения актуальна.

Структура и объем работы: выпускная квалификационная работа выполнена на страницах машинописного текста. Состоит из введения, четырех глав, заключения и приложения.

В первой главе рассматриваются задачи, решаемые с помощью ТПР, описывается суть задачи многокритериального выбора и рассматриваются возможности современных Web-инструментов.

Во второй главе описан процесс разработки системы, производится выбор средств разработки, описана модульная структура и функциональные особенности разрабатываемого ПО, приведено техническое задание на разработку ПО.

В третьей главе демонстрируется проверка работоспособности разрабатываемого программного продукта.

В четвертой главе приведены расчеты экономических параметров проекта.

В заключении, по итогам проделанной работы, сформулированы выводы.

В приложении представлены исходные коды разрабатываемого ПО.

1. ОБЗОР И АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Что такое теория принятия решений

Современные технологии позволяют машинам принимать и обрабатывать информацию любого объема и сложности. ЭВМ во много раз превосходит по скорости вычислений человека. Поэтому люди пишут все больше сложных программ для вычисления, сортировок и других задач, связанных с большим количеством информации.

В данной ВКР разработана онлайн программа, для определения рейтингов сложных объектов по комплексу показателей. В основе программы лежит теория принятия решений. Онлайн калькулятор позволит ранжировать различные объекты по выбранным пользователем показателям несколькими алгоритмами.

Решение задач многокритериального выбора является сложным, трудоемким и ресурсоемким процессом, поэтому разработка инструментов для автоматизации их решения актуальна.

Решение – это выбор определённого сочетания цели, действий, направленных на достижение этой цели, и способов использования имеющихся ресурсов. В рамках социально-экономических систем решение – это результат анализа, прогнозирования, оптимизации и выбора альтернативы из множества вариантов достижения конкретной цели. В узком смысле принятие решений – это заключительный акт анализа вариантов, результат выбора. В широком смысле – это процесс, протекающий во времени. Это совокупность всех этапов и стадий по подготовке решения, включая этап непосредственного принятия решения. Существуют некоторые общие признаки, позволяющие классифицировать решения.

1. По степени повторяемости проблемы:

1.1. Традиционные (неоднократно встречающиеся в практике). Решения здесь – выбор из имеющихся альтернатив.

1.2. Нетипичные (нестандартные). Поиск решений здесь связан с генерацией новых альтернатив.

2. По значимости цели:

2.1. Стратегические (самостоятельные).

2.2. Тактические (решения используются в качестве средства достижение цели более высокого порядка).

3. По сфере воздействия:

3.1. Локальные - результат управленческих решений может сказаться на одном или нескольких элементах системы.

3.2. Глобальные – решения влияют на функционировании системы в целом.

4. По длительности реализации:

4.1. Долгосрочные решение – если между принятием решения и завершением его реализации проходит несколько лет.

4.2. Краткосрочные – если срок небольшой.

5. По прогнозируемым последствиям решений:

5.1. Корректируемые - большинство управленческих решений поддаются корректировке в целях устранения отклонений или учета новых факторов. 5.2. Некорректируемые - решения, последствия которых необратимы.

6. По характеру используемой информации, в зависимости от полноты и достоверности информации:

6.1. Детерминированные (принимаемые в условиях определенности)

6.2. Вероятностные (принимаемые в условиях риска и неопределенности). Большинство решений являются вероятностными.

7. По методам разработки решения:

7.1. Формализованные (выполненные с использованием математических методов).

7.2. Неформализованные (основанные на интуиции и здравом смысле). На практике большинство решений носит комбинированный характер, т.е. применяются попеременно формальные процедуры и неформальные методы.

8. По числу критериев выбора:

8.1. Многокритериальные решения - если выбранная альтернатива должна удовлетворять нескольким критериям.

8.2. Однокритериальные – если критерий один.

9. По форме принятия:

9.1. Коллегиальные (такая форма принятия решений снижает оперативность и размывает ответственность, но препятствует грубым ошибкам и злоупотреблениям и повышает обоснованность выбора).

9.2. Единоличные.

10. По способу фиксации решений:

10.1. Документированные.

10.2. Недокументированные. Процесс принятия решений может быть укрупненно подразделен на 2 операции: выработка рекомендаций специалистами по выбору лучшего варианта и принятие окончательного варианта непосредственно лицом, принимающим решение (ЛПР).

Для лиц принимающих решения (ЛПР) задача принятия решений может быть записана в следующем виде: <C, T, P | Сд, П, Ц, О, А, К, f, A*>,

где С – исходная проблемная ситуация;

Т – время для принятия решения;

Р – потребные ресурсы для принятия решения;

Сд – доопределенная проблемная ситуация;

П = (П1,…, Пn) – множество предположений о развитии ситуации в будущем;

Ц = (Ц1,…, Цk) – множество целей, на достижение которых направлено решение;

О = (О1,…, Оl) – множество ограничений;

А = (А1,…, Аm) – множество альтернативных вариантов решений;

К = (К1,…, Кр) – множество критериев выбора наилучшего варианта;

f – функция предпочтения ЛПР (включает объективные критерии К и личные предпочтения ЛПР);

А * - оптимальное решение.

Рассмотрим более подробно элементы задачи принятия решений. Под проблемой в теории принятия решений понимается разница между фактическим и желаемым состоянием объекта принятия решений. Проблема – неудовлетворительное состояние системы или противоречие, требующее разрешения. Проблема всегда связана с определенными условиями и причинами ее возникновения, которые обобщенно называют ситуацией. Совокупные проблемы и ситуации образуют проблемную ситуацию. Проблемная ситуация формулируется как логическое высказывание, в том числе содержащее неопределенность и нечеткость относительно и целевых параметров, и условий внешней и внутренней среды. В зависимости от того, какая часть целей и условий не определена, возможна дальнейшая структуризация проблемной ситуации. После снятия неопределенности может быть сформулирована управленческая задача. Исходная проблемная ситуация содержательна и, если это возможно, обладает совокупностью количественных характеристик. Располагаемое время Т влияет на возможность получения полной достоверной информации о проблемной ситуации, обоснования вариантов и определения последствий их реализации. В качестве ресурсов для нахождения оптимального решения могут служить знания и опыт людей, научно-технический и информационный потенциал организации, финансы, и т.д. На начальной стадии проблемная ситуация может быть определена не полностью, что связано с неполнотой информации, недостаточной аналитической проработкой. В таких условиях проблемная ситуация доопределяется до уровня, достаточного для действий по принятию решений (Сд). Множество предположений (гипотез) П о развитии ситуации в будущем характеризует неопределенность многих факторов и внешних и внутренних условий и реализации принимаемого решения. Для формирования целей и выбора варианта решения необходимо ориентироваться на определенный вариант развития ситуации. Возможна подготовка вариантов решений для различных предположений о развитии ситуации в будущем. Для четкого

определения вариантов устранения проблемной ситуации необходимо сформулировать множество целей Ц. Реальные задачи, как правило, многоцелевые, кроме того, даже единственная цель может быть разбита на подцели. Цель – это главный системообразующий фактор в любой социально-экономической системе. Правильно поставленная цель становится инструментом решения проблемы. Цель – это состояние объекта управления, к достижению которого стремится система. Реализация решений всегда осуществляется в условиях различных ограничений: финансовых, кадровых, правовых. Поэтому необходимо четко сформулировать множество ограничений, которые должны учитываться при принятии решения в конкретной проблемной ситуации. Для достижения множества целей формулируется множество альтернативных вариантов решений, из которых должно быть выбрано единственное оптимальное или приемлемое решение А*. Множество критериев К используется для абсолютной и/или сравнительной оценки вариантов решений. Абсолютную оценку удается получить в редких случаях. В реальных задачах удается осуществить лишь сравнительную оценку решений. В результате осуществляют предварительный выбор лучшего решения, А*. Окончательный выбор наилучшего решения проводится ЛПР на основе функции предпочтения f. Содержание задачи принятия решений в социально-экономических системах позволяет сформулировать ее особенности:

1. Неизвестные элементы задачи (ситуация, цели, ограничения, варианты решения, предпочтения) имеют содержательный характер и только частично определяются количественными характеристиками. Число неизвестных элементов задачи много больше числа известных.

2. Определение неизвестных элементов задачи и нахождение наилучшего решения не всегда может быть формализовано, т.к. нет готовых алгоритмов.

3. Часть характеристик может быть измерена субъективно (приоритеты целей, критериев, вариантов решения).

4. Часто решать задачи принятия решений приходится в условиях неопределенности, и в таких условиях большое значение имеет интуиция ЛПР.

5. Принимаемые решения могут непосредственно затрагивать интересы ЛПР и специалистов-аналитиков, поэтому их личные предпочтения и мотивы могут повлиять на выбор решения.

В данной работе для принятия решений предлагается использовать метод многокритериального решения.

1.2 Классификация типов проблем, решаемых в теории принятия решений

Существуют большие различия в природе изучаемых проблем принятия решения. Эти различия одним из первых заметил Г.Саймон, который предложил удачную классификацию проблем. Согласно этой классификации, проблемы подразделяются на три класса, т.е. в тех случаях, когда существуют адекватные математические модели устройств или процессов и есть опытные данные.

1. Хорошо структурированные или количественно сформулированные проблемы, в которых существенные зависимости выяснены настолько хорошо, что они могут быть выражены в числах или символах, получающих в конце концов численные оценки.

2. Неструктурированные или качественно выраженные проблемы, в которых известен только перечень основных параметров, но количественные связи между ними установить нельзя (нет необходимой информации). Иногда ясно лишь, что изменение параметра в определенных пределах сказывается на решении. В таких случаях структура, понимаемая как совокупность связей между параметром и ролью не определена, и проблема называется неструктурированной.

3. Слабо структурированные или смешанные проблемы, которые содержат как качественные, так и количественные элементы, причем качественные малоизвестные и неопределенные стороны проблем имеют тенденцию доминировать. Согласно этой классификации типичные проблемы можно назвать хорошо структурированными, т.е. существуют реальности,

допускающие строгое количественно описание и определяющие существование единственного очевидного критерия качества. Изучение реальной ситуации может требовать большого труда и времени. Необходимая информация может быть дорогостоящей. Метод «стоимость – эффективность» представляет собой первые попытки сравнения вариантов решений для слабо структурированных проблем. Типичные неструктурированные проблемы: проблема выбора профессии, конкурсного отбора проектов, выработки политики отбора статей в журналах, тендер. Слабоструктурированные и неструктурированные проблемы исследуются в рамках научного направления, называемого принятием решений при многих критериях.

1.3 Задачи многокритериального выбора

Многокритериальная модель задачи принятия решений (ПР) может быть формально представлена в виде кортежа:

<T, S, K, X, F, P, R>, где

T – анализ проблемной ситуации и выявление целей и определение типа задачи

S – множество альтернатив

K – множество критериев

X – множество шкал

F – отображение множества альтернатив на множестве векторных оценок

P – система предпочтений ЛПР

R – решающее правило.

Как слабо структурированные, так и неструктурированные проблемы должны быть приведены к виду, когда они становятся задачами выбора допустимых альтернатив для достижения заданных целей. Поэтому прежде всего необходимо определить цели. Цель – заранее мыслимый результат сознательной деятельности человека или группы людей. «Тому, кто не знает куда плыть, любой ветер – попутный» Сенека. Рациональными или лучшим

выбором является выбор, который лучше всего удовлетворяет целям. Основные типы задач:

1. Линейное упорядочивание альтернатив.

Рисунок 1.1. (Линейное упорядочивание альтернатив)

Эта дуга имеет место, если альтернатива (например S1) доминирует над другой альтернативой (например S2 , т.е. S1>S2).

2. Выделение лучшей альтернативы.

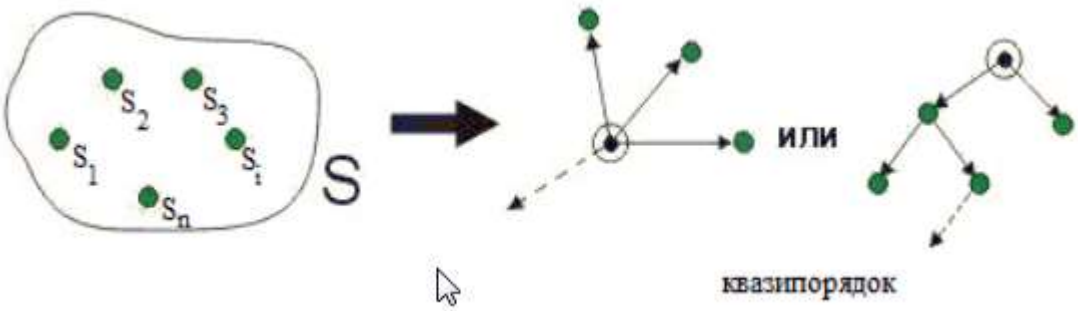


Рисунок 1.2. (Выделение лучшей альтернативы)

Здесь лучшая альтернатива.

3. Выделение неупорядоченного подмножества лучших альтернатив «задача о рюкзаке».

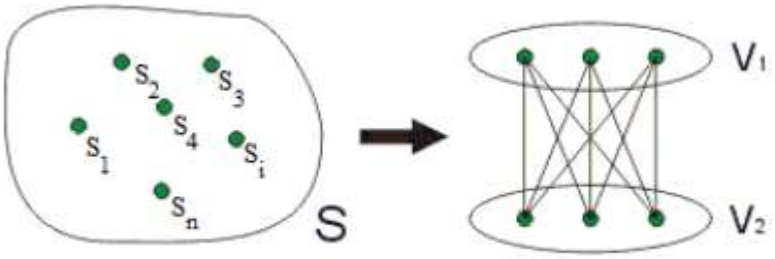


Рисунок 1.3. (задача о рюкзаке)

Здесь V1 - подмножество лучших альтернатив.

4. Групповое упорядочивание альтернатив (стратификация).

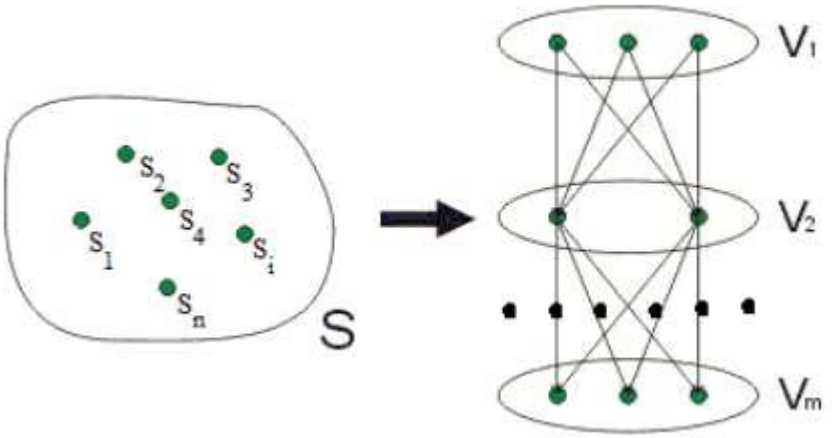


Рисунок 1.4. (Стратификация)

В сложных структурах множество S разбито на непересекающиеся слои: Vi , $i=\overline{1,m}$; $S=\cup i=\overline{1,m}$ Vi ; $\forall(i\neq j)$ Vi ∩ Vj = ∅.

5. Неупорядоченное разбиение альтернатив (классификация)

Здесь Vi – классы (подмножества) альтернатив. Методы структуризации это по существу и есть сердцевина поддержки принятия решений. Задачи принятия решений существенно отличаются в зависимости от требований, которые предъявляются к результатам решения.

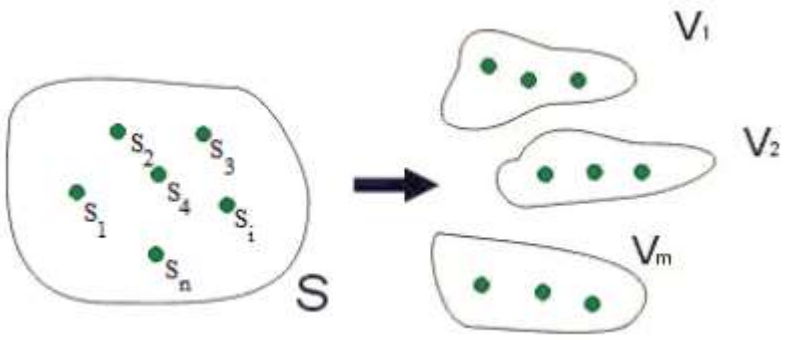


Рисунок 1.5. (Классификация)

Существуют задачи, в которых требуется определить линейный порядок на множестве альтернатив. Так члены семьи упорядочивают по степени необходимости будущие покупки, руководители упорядочивают объекты капиталов вложений. Выпускники вузов упорядочиваются по общим успехам за время обучения и т.д. Часто для решения задачи не нужно иметь совершенный порядок (т.е. линейный порядок). Достаточно иметь квазипорядок, где не все альтернативы сравнимы. При этом часть альтернатив имеет развитый ранг, т.е. их положение в последовательности определенно некоторым интервалом (например, альтернатива хуже 10 и лучше 15 и, следовательно, имеет какой-то ранг в интервале от 11 до 14). Построение квазипорядка требует существенно меньшей информации от ЛПР и в то же время сам квазипорядок может быть удовлетворительным решением для рода практических дуг. При покупке квартиры или дома, при обмене квартиры люди обычно делят альтернативы на две группы: заслуживающие более подробного, требующего затрат и средств изучения, не заслуживающие. Врач, обследующий больных, может выделять группы в соответствии с подразделениями на разные заболевания. Товаровед может группировать товары по качеству. Абитуриент может различать группы вузов по их престижности. Вообще говоря, объединение объектов в группы – очень характерные заметки для людей. Это обосновывается тем, что классификация и стратификация является более удовлетворительным решением для многих практических задач, особенно в том случае, когда число объектов велико. Так, например, не имеет никакого смысла добиваться строгого

ранжирования нескольких сотен объектов. В тоже время разбиение на ряд групп может дать ближе удовлетворительный ответ на вопрос об их качестве.

1.4 Web-приложения как инструмент для принятия решений

Web-приложение — это приложение, работающее на платформе Web, т.е. использующее для взаимодействия с пользователем веб-сервер, работающий по протоколу HTTP и браузер, интерпретирующий страницы HTML. По-другому можно сказать, что это некоторый сайт, содержимое которого изменяется динамически на основе взаимодействия с пользователем.

Хотя возможны различные вариации, чаще всего используется трехуровневая архитектура для построения веб-приложений: веб-браузер, какая-нибудь технология предоставления динамического веб-контента и база данных. Веб-браузер посылает запросы среднему уровню, который обслуживает их, производя запросы к базе данных, и представляя результаты в пользовательском интерфейсе.

Например, рассмотрим самую популярную на сегодняшний день платформу для веб-приложений, т. н. LAMP – Linux, Apache, MySQL, PHP.

Здесь роль среднего уровня играет веб-сервер Apache с установленным на нем модулем поддержки скриптового языка программирования PHP, а в качестве базы данных выступает MySQL.

Достоинства и недостатки, принятого подхода.

Достоинства:

- независимость от клиентской, а часто и серверной платформы;
- нетребовательность к ресурсам клиента (используется т.н. тонкий
- клиент — браузер);
- не требует установки на клиентский компьютер какого-либо программного обеспечения кроме браузера;
- легкость обновления версий веб-приложения, т.к. оно обновляется
- только один раз — на сервере;

- «встроенные» сетевые возможности (возможность работать с
- несколькими клиентами одновременно);
- сохраняются все данные при переходе клиента с машины на машину.

Недостатки и способы их устранения:

- низкое время реакции — ориентация на клиентские технологии, типа JavaScript, DOM, Flash, XUL;
- протокол HTTP не хранит состояния — механизм cookies, сессии;
- малая защищенность – защищенные соединения HTTPS, авторизация встроенная в HTTP или на основе сессий;
- неразвитость языка HTML в смысле форм — разработка стандарта XForms, Flash, Java applets.

1.5 Использование Web-приложений крупными компаниями

Наиболее известные примеры веб-приложений: веб-почта (http://mail.ru), интернет-магазины (http://amazon.com), онлайновые аукционы (http://ebay.com). Однако область применения веб-приложений гораздо шире, чем электронный бизнес, они используются во многих научных и коммерческих областях. Во многих крупных компаниях, таких как банки, веб-приложения используются для организации трудового процесса персонала, например в Сбербанке и банке Хоум-кредит. Также, веб-приложения служат и для предоставления клиентам банков доступа к их услугам, примером такого использования web-приложений может служить система “Сбербанк Онл@йн”. Также стоит отметить, что web-приложения используются и для работы специалистами различных сфер и отраслей, например, компания Oracle, один из крупнейших представителей рынка СУБД и IT сферы в целом имеет, в том числе, и Web-версию приложения для управления БД.

2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ОНЛАЙН КАЛЬКУЛЯТОРА ДЛЯ ОПРЕДЕЛЕНИЯ РЕЙТИНГОВ СЛОЖНЫХ ОБЪЕКТОВ ПО КОМПЛЕКСУ ПОКАЗАТЕЛЕЙ

2.1 Техническое задание

- Онлайн калькулятор для ранжирования сложных объектов по комплексу показалетей (ОКСО)
- Назначение системы: осуществление сравнения групп объектов по ряду показателей.
- Цели создания системы: Программа нужна для того, чтобы оптимизировать процесс решения многокритериальной задачи выбора.
- Требования к системе:
    - Простота интерфейса
    - Кроссплатформенность
    - Низкие требования к аппаратному обеспечению
    - Программа должна принимать csv файл, содержащий данные для сравнения
    - Программа должна предоставлять пользователю выбор параметров, по которым будет осуществляться рассчет рейтингов
    - Программа должна выводить результаты на экран
- Состав и содержание работ:
    - Выбор средств разработки
    - Создание структуры программы
    - Описание модулей
    - Описание спецификаций

2.2 Выбор средств разработки

2.2.1 Язык гипертекстовой разметки Html

World Wide Web - глобальная компьютерная сеть на сегодняшний день содержит миллионы сайтов, на которых размещена всевозможная информация. Люди получают доступ к этой информации посредством использования

технологии Internet. Для навигации в WWW используются специальные программы - Web-браузеры, которые существенно облегчают путешествие по бескрайним просторам WWW. Вся информация в Web-браузере отображается в виде Web-страниц.

Web-страницы, поддерживая технологию мультимедиа, объединяют в себе различные виды информации: текст, графику, звук, анимацию и видео. От того, насколько качественно и красиво сделана та или иная Web-страница, зависит во многом ее успех в Сети.

Пользователю приятно посещать те Web-страницы, которые имеют стильное оформление, не отягощены чрезмерно графикой и анимацией, быстро загружаются и правильно отображаются в окне Web-браузера.

Создать Web-страницу непросто, но наверно каждый человек хотел бы попробовать себя в роли дизайнера.

В своем реферате я сделал попытку разобраться в том, что необходимо знать и уметь для создания Web-страницы, какое программное обеспечение является инструментарием создания Web-страниц и как его эффективно использовать.

Также в данной работе рассмотрены основы языка программирования Web-страниц - HTML, который является общепринятым стандартом WWW. Это даст возможность ознакомиться со структура Web-страницы и приемами ее правильного оформления. А так же при помощи CMS Joomla мы рассмотрим создание сайта.

Web-страницы могут существовать в любом формате, но в качестве стандарта принят Hyper Text Markup Language - язык разметки гипертекстов, предназначенный для создания форматированного текста, насыщенного изображениями, звуком, анимацией, видеоклипами и гипертекстовыми ссылками на другие документы.

Можно работать на Web без знания языка HTML, поскольку тексты HTML могут создаваться разными специальными редакторами и конвертерами. Но писать непосредственно на HTML нетрудно. Возможно, это даже легче, чем

изучать HTML-редактор или конвертер, которые часто ограничены в своих возможностях, содержат ошибки или проводят плохой HTML код, который не работает на разных платформах.

Язык HTML существует в нескольких вариантах и продолжает развиваться, но конструкции HTML скорее всего будут использоваться и в дальнейшем. Изучая HTML и познавая его глубже, создавая документ в начале изучения HTML и расширяя его насколько это возможно, мы имеем возможность создавать Web-страницы, которые могут быть просмотрены многими браузерами Web, как сейчас, так и в будущем. Это не исключает возможности использования других методов, например, метод расширенных возможностей, который предоставляется Opera, Google Chrome, Internet Explorer или другими браузерами.

Работа по HTML - это способ усвоить особенности создания документов в стандартизированном языке, используя расширения, только если это действительно необходимо.

HTML был ратифицирован World Wide Web Consortium. Он поддерживается всеми браузерами.

Поскольку HTML-документы записываются в ASCI I-формате, то для ее создания может использован любой текстовый редактор.

Обычно HTML-документ - это файл с расширением.html или. htm, в котором текст размечен HTML-тегами (англ. tag - специальные встроенные указания). Средствами HTML задаются синтаксис и размещение тегов, в соответствии с которыми браузер отображает содержимое Веб-документа. Текст самих тегов Веб-браузером не отображается.

Все теги начинаются символом '<' и заканчиваются символом '>'. Обычно имеется пара тегов - стартовый (открывающий) и завершающий (закрывающий) тег (похоже на открывающиеся и закрывающиеся скобки в математике), между которыми помещается размечаемая информация:

Информация

Здесь стартовым тегом является тег , а завершающим - . Завершающий тег отличатся от стартового лишь тем, что у него перед текстом в скобках <> стоит символ '/' (слэш).

Браузер, читающий HTML-документ, отображает его в окне, используя структуру HTML-тегов. В каждом HTML-документе должны присутствовать три главных части:

A) Объявление HTML;

B) Заголовачная часть;

C) Тело документа.

A) Объявление HTML

<HTML> и </HTML>. Пара этих тегов сообщает программе просмотра (браузеру) что между ними заключен документ в формате HTML, причем первым тегом в документе должен быть тег <HTML> (в самом начале документа), а последним - </HTML> (в самом конце документа).

<HTML> </HTML>

B) Заголовочная часть.

<HEAD> и </HEAD>. Между этими тегами располагается информация о документе (название, ключевые слова для поиска, описание и т.д.). Однако наиболее важным является название документа, которое мы видим в верхней строке окна браузера и в списках "Избранное (BookMark)". Специальные программы-спайдеры поисковых систем используют название документа для построения своих баз данных. Для того чтобы дать название своему HTML-документу текст помещается между тегами <TITLE> и </TITLE>.

<HTML> <HEAD> <TITLE>Моя первая страница</TITLE> </HEAD> </HTML>

C) Тело документа.

Третьей главной частью документа является его тело. Оно следует сразу за заголовком и находится между тегами <BODY> и </BODY>. Первый из них должен стоять сразу после тега </HEAD>, а второй - перед тегом </HTML>.

Тело HTML-документа - это место, куда автор помещает информацию, отформатированную средствами HTML.

<HTML> <HEAD> <TITLE> Моя первая страница</TITLE> </HEAD> <BODY> .....

</BODY> </HTML>

Теперь мы можем написать HTML-код нашей странички:

<HTML> <HEAD> <TITLE>Моя первая страница</TITLE> </HEAD> <BODY> Здесь будут мои страницы! </BODY> </HTML>

В разделе BODY все символы табуляции и конца строк браузером игнорируются и никак не влияют на отображение страницы. Поэтому перевод строки в исходном тексте HTML-документа не приведет к началу новой строки в отображаемом обозревателем тексте при отсутствии специальных тегов. Это правило очень важно помнить и не забывать ставить разделяющие строки теги, иначе у текста не будет абзацев, и он станет нечитаемым.

Для начала новой строки используется тег
 (сокр. от англ. break - прервать). Этот тег приводит к отображению браузером дальнейшего текста с начала следующей строки. Закрывающий для него тег не используется. Он удобен, если требуется с какого-то места писать с новой строки без начала нового абзаца, например, в стихотворении. Повторное его использование позволяет вставить одну или несколько пустых строк, отодвинув следующий фрагмент страницы вниз.

Сплошной текст без промежутков читается не очень легко, его неудобно просматривать и находить нужные места. Разбитый на абзацы, текст воспринимается гораздо быстрее. Для начала нового абзаца используется тег <p> (англ. paragraph - абзац). Этот тег, кроме начала новой строки, вставляет одну пустую строку. Но многократное повторение <p>, в отличие от <br>, не приведет к появлению нескольких пустых строк, останется все та же одна пустая строка.

Внутри скобок тега кроме его названия могут размещаться также атрибуты (англ. atributes - атрибуты). Они отделяются от названия и между

собой пробелами (одним или несколькими), а пишутся в виде имя_атрибута="значение". Если значение не содержит пробелов, то кавычки могут быть опущены, но так делать не рекомендуется. Тег может содержать атрибут ALIGN, определяющий выравнивание абзаца. По умолчанию абзац выравнен влево ALIGN="left". Возможны также выравнивания вправо ALIGN="right" и по центру ALIGN="center". При использовании атрибутов, после форматируемого текста следует использовать закрывающий тег . Если его нет, то новый тег означает закрытие предыдущего, соответственно вложенные невозможны. Выравнять текст по центру возможно также тегом <CENTER>.

Теперь можно поместить на Web-страницу некоторый текст с различным выравниванием:

<HTML> <HEAD> <TITLE>Моя первая страница</TITLE> </HEAD> <BODY> Здесь будут мои личные страницы! На них Вы сможете найти: <bp> - рассказ обо мне и о моих увлечениях; <bp> - мои фотографии.

С одной из моих страниц можно будет
отправить мне электронное письмо.

</BODY> </HTML>

Каждый выбирает свой инструмент для создания Web-страниц. Это может быть MS FrontPage или Macromedia DreamWeaver, Allaire HomeSite или 1st Page. А кто-то пользуется простым текстовым редактором, например Блокнотом (Notepad).

Текстовые редакторы возможно использовать только для создания небольших страниц, так как у них есть много минусов: не поддерживаются проекты, отсутствует "подсветка" текста., в общем, работать крайне неудобно.

Основным недостатком MS FrontPage является то, что он генерирует очень большой HTML-код (слишком много лишнего), поэтому страницы получаются большими, что сказывается на скорости загрузки. Более того, при создании Web-страниц в этом редакторе видишь одно, а в окне браузера -

совсем другое. Странички получаются какими-то кривыми, поэтому для создания качественных Web-страниц рекомендуется использовать пакеты, которые будут рассмотрены ниже.

Начнем с популярного Macromedia DreamWeaver. Компания Macromedia считается лидером по производству программ для создания веб-сайтов, а также законодателем моды в этой области.

DreamWeaver 3.0

Но DreamWeaver на несколько шагов опережает другие редакторы, использующие технологию WYSIWYG, в первую очередь тем, что генерирует очень чистый HTML-код. DreamWeaver позволяет вам избавиться от однотипной работы при создании страниц (например, верстка текста) при помощи использования опции "запись последовательности команд" (вы записываете последовательность производимых команд, потом нажимаете, например, CTRL+P, и DreamWeaver воспроизводит все в той же последовательности.

HomeSite 4.0

Следующий редактор - HomeSite 4 - для создания страниц вручную, т.е. для знатоков HTML. Вы получаете полный контроль над HTML-кодом, причем существует возможность оптимизировать свою страничку под один из трех популярных браузеров (MSIE, Google Phrome, Opera).

HomeSite содержит два основных режима: Edit и Design. Режим Design - это подобие WYSIWYG-редактора, выдающее HTML-код, причем, если вы загрузите чужой HTML-код, то HomeSite все перепишет по-своему.

Еще одна отличительная особенность HomeSite - это его "склейка" с Dreamweaver. HomeSite обладает кнопкой "Dreamweaver", а также входит в его стандартный пакет поставки. Впрочем, и DreamWeaver имеет возможность подключения HomeSite, как редактора для корректировки HTML-кода.

Одним из последних HTML-редакторов является EVR Soft 1st Page v2.

Его лозунг - "Create 1st class websites!" ("Создавайте первоклассные веб-сайты!"). Редактор содержит несколько режимов - Normal, Easy,

Advanced/Expert и Hardcore, то есть вы можете выбрать свой уровень, а со временем перейти на более высокий. Еще одна особенность - довольно большая коллекция скриптов на JavaScript и DHTML. Все это довольно удобно разбито по категориям.

Создание и оптимизация графики - сложная задача. Безусловно, возможно создание Web-страницы и без использования графики - при помощи шрифтов, скриптов и таблиц стилей (CSS) - и это будет красиво и стильно. Но ведь окончательный вид документа зависит от большого числа различных факторов, таких как: ширина окна браузера, предварительные настройки браузера, принятые по умолчанию размер шрифта, его имя и цвет. К тому же не все скрипты и стили поддерживаются всеми браузерами. Если же будет использована графика, то посетитель вашей страницы увидит ее точно такой, какой сделали и видите ее вы.

Основная сложность работы с Web-графикой состоит в том, что пропускная способность каналов Интернета, в большинстве случаев, очень низкая и перед вами сразу встанут проблемы - как сделать графический файл небольшой по объему, но хорошего качества, какие программы и приемы использовать при его оптимизации.

Именно этому посвящен раздел о векторные и растровые графических редакторах, которые являются мощным инструментом обработки изображения в умелых руках.

Прежде чем рассмотреть векторные и растровые графические редакторы, следует уяснить себе, в чем состоит различие между векторным и растровым представлением изображения.

Растровая графика представляет собой сетку (растр), ячейки которой называются пикселами. Каждый пиксел в растровом изображении имеет строго определенное местоположение и цвет, следовательно любой объект представляется программой как набор окрашенных пикселов. Это значит, что пользователь, работая с растровыми изображениями, работает не над конкретными объектами, а над составляющими их группами пикселов.

Растровые изображения обеспечивают высокую точность передачи градаций цветов и полутонов, а также высокую детализацию изображения, поэтому они являются оптимальным средством представления тоновых изображений, таких как сканированные фотографии.

Для изображения растровой графики всегда используется фиксированное количество пикселов, т.е. качество растрового изображения напрямую зависит от разрешающей способности оборудования. Это значит, что любое изменение изображения (поворот, увеличение и т.д.) приводит к неизменному искажению картинки и границы объектов получаются неровными.

Векторные изображения формируются на основе математически описанных фигур, называемых векторами, а вид изображения определяется параметрами векторов. Другими словами, векторная графика состоит из кривых, имеющих координаты, цвет и прочие параметры, а также замкнутых областей, заполненных определенным цветом. Границы этих областей также описываются кривыми. Файл с векторной картинкой содержит координаты и параметры кривых.

Результаты обработки векторных изображений не зависят от разрешающей способности оборудования, поэтому вы можете произвольно изменять их параметры (размер, цвет, форму и т.д.) - качество не ухудшится. Векторная графика применяется при создании цифровых объектов с использованием мелких кеглей (размеров шрифта) или таких объектов, как логотипы, для которых важно сохранять четкие контуры, при неограниченном масштабировании.

Графические пакеты (редакторы) тоже делятся на два типа: растровые и векторные. Рассмотрим наиболее популярные из них.

А) Редакторы растровой графики

Язык гипертекстовой разметки HTML.

Microsoft Paint - простой (или лучше сказать - простейший) редактор, входящий в стандартную поставку операционных систем Microsoft. Он

обладает набором простейших функций (кисточка, карандаш, резинка и т.д.), которые позволяют создавать незамысловатые картинки. К сожалению, для обработки графики он практически не пригоден. Картинку, которую вы видете справа - это большее, на что способен этот редактор.

Adobe Photoshop - на сегодняшний день это самый мощный пакет для профессиональной обработки растровой графики. Это целый комплекс, обладающий многочисленными возможностями модификации растрового рисунка, имеющий огромный набор различных фильтров и эффектов, причем есть возможность подключать инструменты независимых производителей.

Пакет предлагает, например, средства для восстановления поврежденных изображений, ретуширования фотографий или создания самых фантастических коллажей, которые только может позволить себе наше воображение. В общем, потенциал этого пакета поистине огромен. Начиная с версии 5.5 в пакет включена программа Adobe ImageReady, предоставляющие огромные возможности по обработке графики под WEB (оптимизация изображений, создание анимированных gif, "разрезание" картинок на более мелкие и т.д.). Девиз разработчиков Adobe Photoshop - "Camera of your mind" - предполагает не только техническое совершенство, но и полную свободу творчества, на которую человек, работающий с этой программой, просто обречен.

PhotoPaint - еще один не менее известный графический редактор (из пакета Corel Draw) для обработки растровой графики, конкурирующий с Adobe Photoshop. Здесь также имеются все необходимые инструменты для обработки графики, разнообразные фильтры, текстуры. Разница лишь в удобстве работы, интерфейсе и скорости наложения фильтров - наложение происходит немного медленнее.

Painter - редактор предоставляет великолепные возможности для эмуляции реальных инструментов рисования: графит, мел, масло и т.д. Также позволяет имитировать фактуру поверхности материалов, живопись, создавать анимацию. Очень удобен для разработки фоновых рисунков или Web-страниц в

стиле живописи. Пользуясь это программой чувствуешь себя настоящим художником.

Существует еще ряд редакторов (Microsoft Photo Editor, Microsoft Photo DRAW), также позволяющих реализовать простейшие задачи, но не удовлетворяющих запросам профессионалов.

В) Редакторы векторной графики

Язык гипертекстовой разметки HTML

Adobe Illustrator - пакет позволяет создавать, обрабатывать и редактировать векторную графику. По своей мощности он эквивалентен

растровому редактору Adobe Photoshop: имеет аналогичный интерфейс, позволяет подключать различные фильтры и эффекты, понимает многие графические форматы, даже такие как. cdr (Corel Draw) и. swf (Flash).

CorelDraw - безусловно, такой известный графический пакет не мог обойтись без средств для обработки векторной графики. Пакет по своей мощности практически не уступает графическим редакторам Adobe Photoshop и Adobe Illustrator. Помимо обработки векторной графики, в этом пакете существует обработчик растровой графики (Photo Paint), трассировщик изображений, редактор шрифтов, подготовки текстур и создания штрихкодов, а также огромные коллекции с изображениями (CorelGallery).

Adobe Streamline - еще один продукт фирмы Adobe, предназначенный для трассировки (перевода) растровой графики в векторную. Это небольшой, но очень полезный и мощный продукт. Особенно полезен, если вы создаете Web-страницы с использованием векторной графики, например, технологии Flash.

2.2.2 Выбор языка для разработки

PHP - язык программирования, используемый на стороне WEB-сервера для динамической генерации HTML-страниц. Об этом говорит и расшифровка его названия: PHP - Personal HyperText Processor.

PHP - один из немногих языков программирования, созданных специально для разработки веб-приложений. Поэтому он включает в себя все функции, необходимые именно для работы на веб-сервере, и при этом лишен избыточности, свойственной многим его конкурентам.

Очень приятная особенность PHP - то, что его команды включаются в обычные HTML-страницы с помощью специальных тегов, которые и заставляют PHP-машину выполнять на сервере нужные действия. Программам на PHP не нужны специальные CGI-директории с особыми правами доступа. Более того, на одной страничке можно произвольно чередовать "простой" HTML и PHP-код.

PHP не зависит от платформы. PHP прекрасно интегрируется во все популярные веб-серверы: Apacce и IIS, Zens и Netscape Enterprise Server, работает под Windows и OS/2, MacOS и практически всеми UNIX-подобными системами. Как следствие - PHP работает практически у всех хостеров, разрешающих собственные выполняемые скрипты.

Замечательная особенность PHP - его интегрированность практически со всеми современными интернет-технологиями. PHP поддерживает большинство современных веб-протоколов: IMAP, FTP, POP, XML, SNMP и другие. PHP прекрасно работает с базами данных. Трудно найти СУБД, поддержка которой не была бы реализована в PHP. MySQL и MS SQL Server, PostgreSQL и Oracle, Sybase и Interbase... Один только перечень баз данных, поддерживаемых PHP, займет, наверное, целый экран.

PHP включает в себя огромное количество встроенных функций: обработки строк и массивов, работы с файловой системой и с HTTP, электронной почтой, датой и временем, кириллицей и другими национальными алфавитами. Когда я впервые начал программировать на PHP, то был просто поражен обилием встроенных функций! Благодаря им многие алгоритмы, требующие в большинстве языков написания программного кода размером в несколько экранов, реализуются на PHP одной командой (точнее, вызовом одной функции).

Современные тенденции развития языков программирования не обошли стороной и PHP. Средства объектно-ориентированного программирования появились еще в PHP3. А в объектной модели PHP4 в полном объеме реализованы классические понятия объектно-ориентированного программирования: наследование, инкапсуляция и полиморфизм.

PHP – это скрипт-язык (scripting language), встраиваемый в HTML, который интерпретируется и выполняется на сервере.

Основное отличие от CGI-скриптов, написанных на других языках, типа Perl или C – это то, что в CGI-программах вы сами пишете выводимый HTML-код, а, используя PHP – вы встраиваете свою программу в готовую HTML-страницу, используя открывающий и закрывающий теги (в примере <?php и ?>).

Отличие PHP от JavaScript, состоит в том, что PHP-скрипт выполняется на сервере, а клиенту передается результат работы, тогда как в JavaScript-код полностью передается на клиентскую машину и только там выполняется.

Любители Internet Information Server найдут, что PHP очень похож на Active Server Pages (ASP), а энтузиасты Java скажут, что PHP похож на Java Server Pages (JSP). Все три языка позволяют размещать код, выполняемый на Web-сервере, внутри HTML страниц.

В нескольких словах – на PHP можно сделать все, что можно сделать с помощью CGI–программ. Например: обрабатывать данные из форм, генерировать динамические страницы, получать и посылать куки (cookies).

Кроме этого в PHP включена поддержка многих баз данных (databases), что делает написание Web-приложений с использованием БД до невозможности простым.

Вот неполный перечень поддерживаемых БД:

Adabas D, InterBase, Solid, dBasemSQL, Sybase, Empress, MySQL, Velocis, FilePro, Oracle, Unix dbm, Informix, PostgreSQL.

Вдобавок ко всему PHP понимает протоколы IMAP, SNMP, NNTP, POP3 и даже HTTP, а также имеет возможность работать с сокетами (sockets) и общаться по другим протоколам.

Разработчикам Web-приложений нет необходимости говорить, что web-страницы - это не только текст и картинки. Достойный внимания сайт должен поддерживать некоторый уровень интерактивности с пользователем : поиск информации, продажа продуктов, конференции и т.п. Традиционно все это реализовалось CGI-скриптами, написанными на Perl. Но CGI- скрипты очень плохо масштабируемы. Каждый новый вызов CGI, требует от ядра порождения нового процесса, а это занимает процессорное время и тратит оперативную память. PHP предлагает другой вариант – он работает как часть Web-сервера, и этим самым похож на ASP от Microsoft.

Синтаксис PHP очень похож на синтаксис C или Perl. Люди, знакомые с программированием, очень быстро смогут начать писать программы на PHP. В этом языке нет строгой типизации данных и нет необходимости в действиях по выделению/освобождению памяти.

Программы, написанные на PHP, достаточно легкочитаемы. Написанный PHP – код легко зрительно прочитать и понять, в отличие от Perl-программ.

Недостатки PHP:

- PHP является интерпретируемым языком, и, вследствие этого, не может сравниться по скорости с компилируемым C. Однако при написании небольших программ, что, в общем-то, присуще проектам на PHP, когда весь проект состоит из многих небольших страниц с кодом, вступают в силу накладные расходы на загрузку в память и вызов CGI-программы, написанной на C;
- не такая большая база готовых модулей, как, например, CPAN у Perl. С этим ничего нельзя поделать – это дело времени. В PHP 4 разработчики предусмотрели специальный репозиторий PEAR, аналогичный CPAN, и я думаю, очень скоро будет написано достаточное количество модулей для его наполнения.

Java Script

Гипертекстовая информационная система состоит из множества информационных узлов, множества гипертекстовых связей, определенных на этих узлах и инструментах манипулирования узлами и связями. Технология World Wide Web - это технология ведения гипертекстовых распределенных систем в Internet, и, следовательно, она должна соответствовать общему определению таких систем. Это означает, что все перечисленные выше компоненты гипертекстовой системы должны быть и в Web.

Web, как гипертекстовую систему, можно рассматривать с двух точек зрения. Во-первых, как совокупность отображаемых страниц, связанных гипертекстовыми переходами (ссылками - контейнер ANCHOR). Во-вторых, как множество элементарных информационных объектов, составляющих отображаемые страницы (текст, графика, мобильный код и т.п.). В последнем случае множество гипертекстовых переходов страницы - это такой же информационный фрагмент, как и встроенная в текст картинка.

При втором подходе гипертекстовая сеть определяется на множестве элементарных информационных объектов самими HTML-страницами, которые и играют роль гипертекстовых связей. Этот подход более продуктивен с точки зрения построения отображаемых страниц "на лету" из готовых компонентов.

При генерации страниц в Web возникает дилемма, связанная с архитектурой "клиент-сервер". Страницы можно генерировать как на стороне клиента, так и на стороне сервера. В 1995 году специалисты компании Netscape создали механизм управления страницами на клиентской стороне, разработав язык программирования JavaScript.

Таким образом, JavaScript - это язык управления сценариями просмотра гипертекстовых страниц Web на стороне клиента. Если быть более точным, то JavaScript - это не только язык программирования на стороне клиента. Liveware, прародитель JavaScript, является средством подстановок на стороне сервера Netscape. Однако наибольшую популярность JavaScript обеспечило программирование на стороне клиента.

Основная идея JavaScript состоит в возможности изменения значений атрибутов HTML-контейнеров и свойств среды отображения в процессе просмотра HTML-страницы пользователем. При этом перезагрузки страницы не происходит.

На практике это выражается в том, что можно, например, изменить цвет фона страницы или интегрированную в документ картинку, открыть новое окно или выдать предупреждение.

Название "JavaScript" является собственностью Netscape. Реализация языка, осуществленная разработчиками Microsoft, официально называется Jscript. Версии JScript совместимы (если быть совсем точным, то не до конца) с соответствующими версиями JavaScript, т.е. JavaScript является подмножеством языка JScript.

JavaScript стандартизован ECMA (European Computer Manufacturers Association - Ассоциация европейских производителей компьютеров). Соответствующие стандарты носят названия ECMA-262 и ISO-16262. Этими стандартами определяется язык ECMAScript, который примерно эквивалентен JavaScript 1.1. Отметим, что не все реализации JavaScript на сегодня полностью соответствуют стандарту ECMA. В рамках данного курса мы во всех случаях будем использовать название JavaScript.

Для создания механизма управления страницами на клиентской стороне было предложено использовать объектную модель документа. Суть модели в том, что каждый HTML-контейнер - это объект, который характеризуется тройкой:

- свойства;
- методы;
- события.

Объектную модель можно представить как способ связи между страницами и браузером. Объектная модель - это представление объектов, методов, свойств и событий, которые присутствуют и происходят в программном обеспечении браузера, в виде, удобном для работы с ними кода

HTML и исходного текста сценария на странице. Мы можем с ее помощью сообщать наши пожелания браузеру и далее - посетителю страницы. Браузер выполнит наши команды и соответственно изменит страницу на экране.

Объекты с одинаковым набором свойств, методов и событий объединяются в классы однотипных объектов. Классы - это описания возможных объектов. Сами объекты появляются только после загрузки документа браузером или как результат работы программы. Об этом нужно всегда помнить, чтобы не обратиться к объекту, которого нет.

Преимущества JavaScript:

- ни один современный браузер не обходится без поддержки JavaScript;
- с использованием написанных на JavaScript плагинов и скриптов справится даже не специалист;
- полезные функциональные настройки;
- постоянно совершенствующийся язык – сейчас разрабатывается бета-вариация проекта, JavaScript2;
- взаимодействие с приложением может осуществляется даже через текстовые редакторы – Microsoft Office и Open Office;
- перспектива использования языка в процессе обучения программированию и информатике.

Недостатки JavaScript.

- пониженный уровень безопасности ввиду повсеместного и свободного доступа к исходным кодам популярных скриптов;
- множество мелких раздражающих ошибок на каждом этапе работы. - Большая часть из них легко исправляется, но их наличие позволяет считать этот язык менее профессиональным, сравнительно с другими;
- повсеместное распространение. Своеобразным недостатком можно считать тот факт, что часть активно используемых программ (особенно приложений) перестанут существовать при отсутствии языка, поскольку целиком базируются на нем.

Сравнив оба языка для Web-программирования для разработки приложения ВКР был выбран язык js, ввиду отсутствия необходимости большой вычислительной мощности для работы программы.

2.3 Модульная структура программы

2.3.1 Общее описание модулей

Код программы “онлайн калькулятор для определения рейтингов сложных объектов по комплексу показателей” имеет модульную структуру и состоит из следующих частей:

- модуль ввода-вывода - содержит методы чтения данных из файла и вывода результата;
- модуль вычислений - содержит вычислительные функции;
- модуль интерфейса - функции проверки состояния checkboxes и запуск вычислений и ранжирования объектов.

Классы, содержащиеся в модулях, спроектированы и разработаны в соответствии с гибкой стратегией разработки и соответствуют принципам SOLID. Это позволило сделать модули максимально независимыми друг от друга, что упрощает процесс их внедрения в систему, а так же делает процесс тестирования и редактирования модулей более простым.

2.3.2 Спецификации подпрограмм

Модуль ввода-вывода:

pars

- назначение: формирует массив из принятого файла;
- входные данные: файл типа csv;
- выходные параметры: массив данных.

vivod

- назначение: показывает результаты вычислений;
- входные данные: массив данных;
- выходные параметры: отсутствуют.

Модуль вычислений

start

- назначение: создает массив выбранных параметров;
- входные данные: массив данных;
- выходные параметры: массив данных.

main

- назначение: рассчитывает входные параметры и передает результаты в массив;
- входные данные: массив данных;
- выходные параметры: массив данных.

sort

- назначение: сортирует принятый массив;
- входные данные: массив данных;
- выходные параметры: массив данных.

Модуль интерфейса

- вычислить
- назначение: запускает вычислительный модуль

check

- назначение: определяет, какой чекбокс включен и не позволяет включить более одного;
- входные данные: массив значений чекбоксов;
- выходные параметры: отсутствуют.

check2

- назначение: определяет, какие чекбоксы включены;
- входные данные: массив значений чекбоксов;
- выходные параметры: массив включенных чекбоксов.

2.4 Описание функциональности системы в виде пользовательских сценариев

Прецедент "Выбор параметров для ранжирования"

Цель: Выбор верного количества параметров, по которым будут ранжироваться объекты

Активаторы: кнопка «вычислить».

Основной успешный сценарий: вывод на экран результатов ранжирования.

Альтернативный сценарий: Информирование об ошибке (Выберите 3 параметра).

Прецедент "Дополнительные параметры"

Цель: При выборе одного из условий появляется возможность выбора дополнительных параметров для ранжирования

Активаторы: чекбокс.

Основной успешный сценарий: вывод на экран дополнительных чекбоксов.

Альтернативный сценарий: -

3. ТЕСТИРОВАНИЕ И ПРОВЕРКА РАБОТОСПОСОБНОСТИ ОНЛАЙН КАЛЬКУЛЯТОРА.

Тестирование прецедента «Дополнительные параметры»:

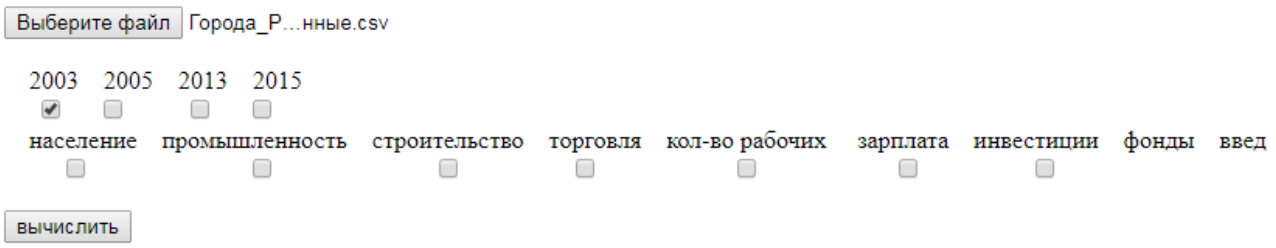


Рисунок 3.1 (Прецедент 1.1)

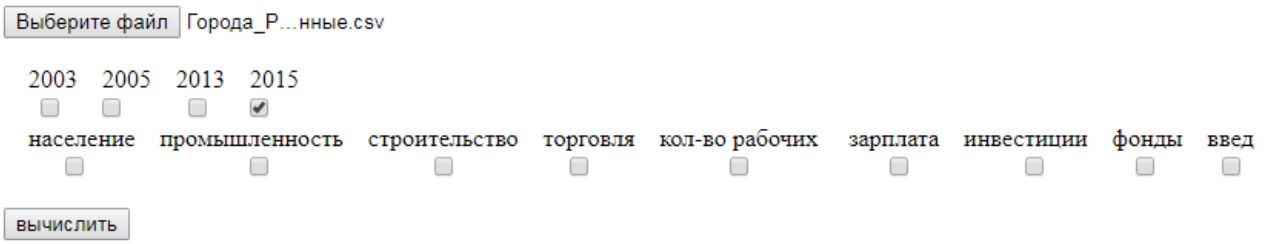


Рисунок 3.2 (Прецедент 1.2)

Тестирование прецедента «Выбор параметров для ранжирования»:

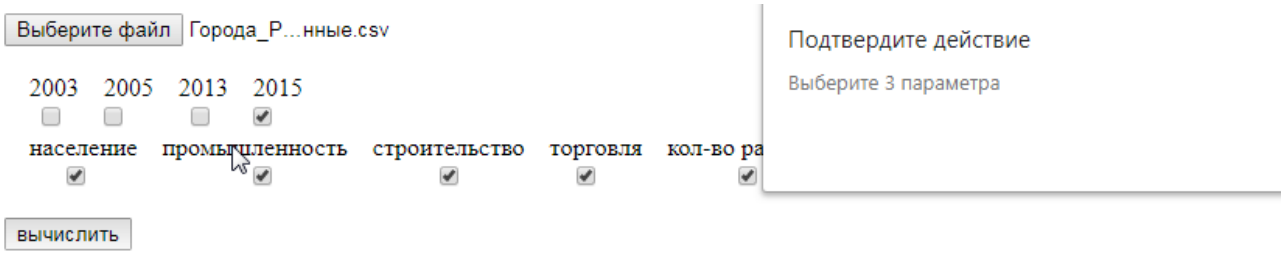
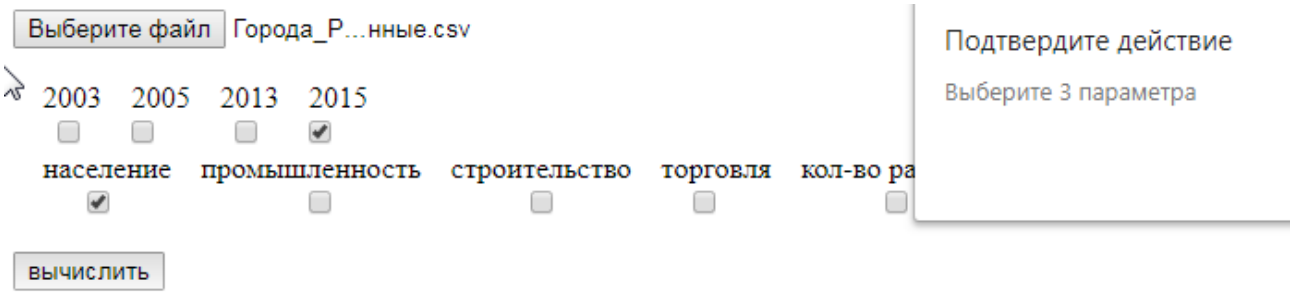


Рисунок 3.3 (Прецедент 2.1)



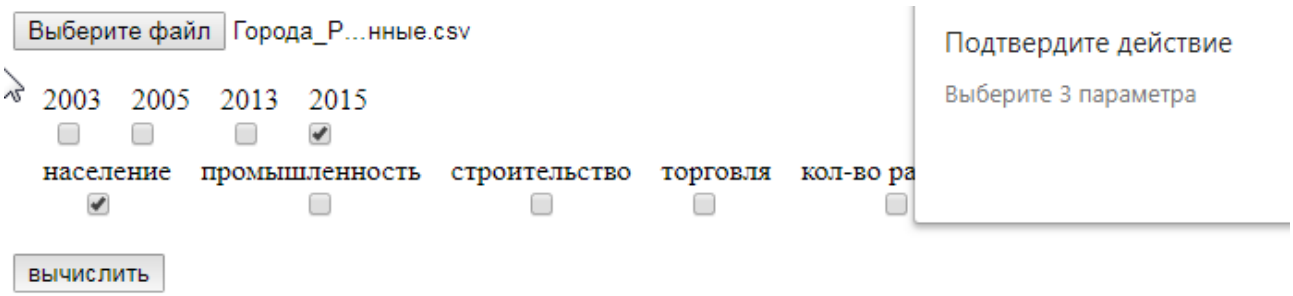


Рисунок 3.4 (прецедент 2.2)

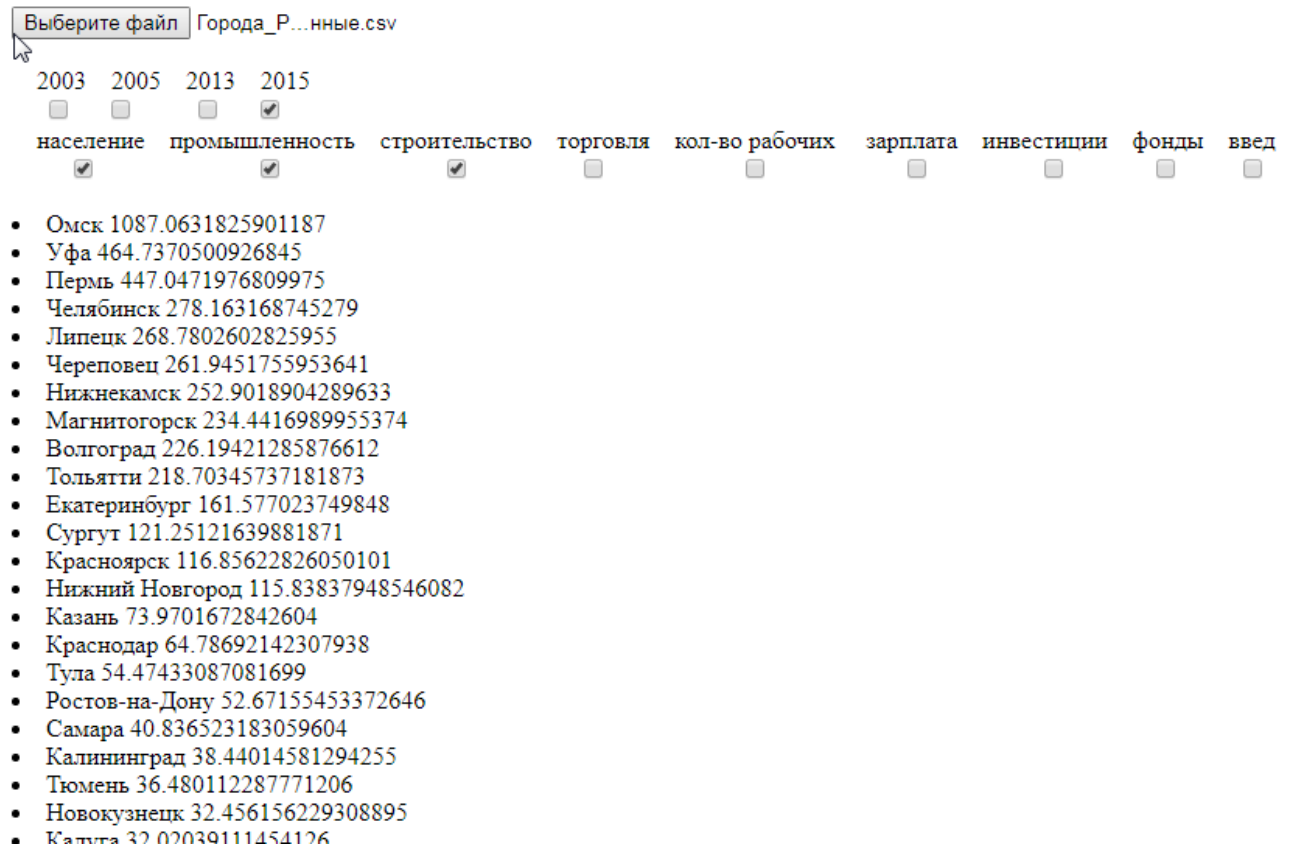


Рисунок 3.5 (Прецедент 2.3)

4. ЭКОНОМИЧЕСКАЯ ЭФФЕКТИВНОСТЬ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

4.1. Технико-экономическое обоснование разработки программного обеспечения

Задачи многокритериального выбора возникают перед каждым человеком и большей части организаций. Наличие решений, способных быстро и точно произвести решение подобных задач способно вывести продуктивность людей, занимающихся их решением на новый, более быстрый и качественный уровень. При разработке такого программного обеспечения крайне важным пунктом является простота в освоении и использовании и точность вычислений. В случае, если решение окажется сложным в эксплуатации, организациям придется тратить большие объемы ресурсов на освоение сотрудниками данных инструментов.

4.2. Расчет единовременных затрат на разработку ПО

К единовременным затратам относят затраты на теоретические исследования, постановку задачи, проектирование, разработку алгоритмов и программ, отладку, опытную эксплуатацию, оформление документов, исследование рынка и рекламу. Фактическая трудоемкость по стадиям проектирования представлена в таблице 4.1.

Таблица 4.1 – Содержание стадий НИР

Этапы	Трудоёмкость	
	Дни	%
Техническое задание	20	15.38

Таблица 4.1 - Продолжение

Эскизный проект	30	23.08
Технический проект	20	15.38
Рабочий проект	30	23.08
Внедрение	30	23.08
ИТОГО	**130**	**100**

В смету затрат на разработку ПО включают:

- материальные затраты;
- основная и дополнительная зарплата разработчика, руководителя и консультанта;
- отчисления на социальные нужды;
- стоимость машинного времени на подготовку и отладку программ;
- стоимость инструментальных средств;
- затраты на Интернет ресурсы;
- накладные расходы.

4.2.1. Материальные затраты

Под материальными затратами понимают стоимость всех материалов, использующиеся в процессе разработки и внедрения ПО (в том числе стоимость бумаги, красящей ленты или картриджа, оплата услуг интернета, дискет и т.д.), в действующих ценах. Данные о материальных затратах на данный проект представлены в таблице 4.2.

Таблица 4.2 — Материальные затраты

Наименование	Количество, шт.	Цена за единицу, руб.	Всего, руб.
Интернет	5	800,00	4000,00
ИТОГО			**4000,00**

4.2.2. Основная и дополнительная зарплата

Основная заработная плата З включает заработную плату разработчика ПО, руководителя проекта и консультанта:

$$З = Т_{об} \times З_{ср.дн} \ , \quad (4.1)$$

где $З_{ср.дн}$ — среднедневная зарплата одного работника, руб.;

$Т_{об}$ — общая трудоемкость проекта, дни.

Для разработчика ПО трудоемкость, согласно данным таблицы 4.1, составляет 130 дней (780 часов), для руководителя – 48 часов, для консультанта – 18 часов.

Дополнительная зарплата рассчитывается как 10% к основной заработной плате. Данные о заработной плате работников представлены в таблице 4.3.

Таблица 4.3 — Заработная плата работников

Работник	Трудоёмкость	Ставка, руб./час	Основная з/п, руб.	Дополнительная з/п, руб.
Разработчик	780	100	78000	7800
Руководитель	48	100	4800	480
Консультант	18	100	1800	180
ИТОГО			**84600**	**8460**

4.2.3. Отчисления на социальные нужды

Отчисления на социальные нужды по действующему законодательству на момент выполнения дипломного проекта составляют 30%:

$$О_{соц.н.} = (З_{осн.} + З_{доп.}) \times 0{,}3 \quad (4.2)$$

Следовательно, отчисления на социальные нужды составляют:

$$О_{соц.н.} = (84600+8460) \times 0,3 = 27918 \text{ руб.}$$

4.2.4. Стоимость машинного времени на подготовку и отладку ПО

Стоимость машинного времени $З_{маш.вр}$ зависит от себестоимости машино-часа работы ЭВМ и времени работы ЭВМ и включает амортизацию ЭВМ и оборудования и затраты на электроэнергию.

Рассчитываем амортизацию следующим образом:

$$А_м = О_ф \times Н_{ам} 365 \times 100 \times Т_м, \quad (4.3)$$

где $О_ф$ — стоимость ЭВМ и оборудования (основные фонды);

$Н_{ам}$ — норма амортизации, принять равной 30%;

$А_м$ — амортизационные отчисления, руб.;

$Т_м$ — время использования оборудования, дни, равное:

$$Т_м = 0,35 \times Т_{эск} + 0,6 \times Т_{тех.пр} + 0,8 \times Т_{раб.пр} + 0,6 \times Т_{вн}, \quad (4.4)$$

где $Т_{эск}$, $Т_{тех.пр.}$, $Т_{раб.пр.}$, $Т_{вн}$ — затраты времени на разработку эскизного, технического, рабочего проекта и внедрение соответственно, в днях:

$$Т_м = 0,35 \times 30 + 0,6 \times 20 + 0,8 \times 30 + 0,6 \times 30 = 64,5 \text{ дн.}$$

Данные об основных фондах представлены в таблице 4.4.

В таком случае амортизация будет равна:

$$А_м = 35000 \times 30 / 36500 \times 64,5 = 1855,47 \text{ руб.}$$

Таблица 4.4 — Основные фонды

Наименование	Количество, шт.	Цена за единицу, руб.	Всего, руб.
ЭВМ	1	34890,00	34890,00
Итого			**34890,00**

Затраты на электроэнергию

$$З_{эл} = С_{эл} \times М_{эвм} \times Т_{м} \times Т_{сут} \quad , \qquad (4.5)$$

где $С_{эл}$ — стоимость 1 кВт/ч электроэнергии, руб.;

$М_{эвм}$ — мощность ЭВМ, кВт/ч;

$Т_{сут}$ — время рабты ЭВМ в сутки, час.

$$З_{эл}=3,74 \times 0,08 \times 64,5 \times 6=115,79 \text{ руб.}$$

$$З_{маш.вр}=А_{м}+З_{эл}=1971,26 \text{ руб.}$$

4.2.5. Стоимость инструментальных средств

Стоимость инструментальных средств $А_{ис}$ включает стоимость системного программного обеспечения, использованного при разработке проекта в размере износа за этот период (таблица 4.5).

Таблица 4.5 — Расчет затрат на инструментальные средства

Наименование	Количество, шт.	Цена за единицу, руб.	Всего, руб.
ОС Windows 10 Домашняя	1	7829,00	7829,00
Итого			**7829,00**

Норма амортизации: $Н_{ам}$=30%

Стоимость инструментальных средств:

$$А_{ис} = 7829 \times 30/36500 \times 64,5 = 415.05 \text{ руб.}$$

4.2.6. Накладные расходы

Накладные расходы определяют в размере 20% от основной заработной платы:

$$Р_н = З_{осн} \times 0,2, \quad (4.6)$$

$$Р_н = 84600 \times 0,2 = 16920 \text{ руб.}$$

4.2.7. Смета затрат на разработку ПО

Используя все вышеприведенное, составим смету затрат (таблица 4.6).

Таблица 4.6 – Смета затрат на разработку ПО

Элементы затрат	Стоимость, руб.
Материальные затраты	4000,00
Основная и дополнительная зарплата	93060,00
Затраты на оплату машинного времени	1971,26
Стоимость инструментальных средств	415,05
Накладные расходы	16920,00
Итого	**116366,31**

Распределение инвестиций по времени реализации проекта осуществляется на основе итоговой суммы затрат на разработку (таблица 4.7).

Таблица 4.7 — Процент времени от общего числа по каждой части проекта и соответствующий объем инвестиций

Тип работы	Трудоемкость, %	Объем инвестиций
Техническое задание	15,38	17897,13
Эскизный проект	23,08	26857,35
Технический проект	15,38	17897,13
Рабочий проект	23,08	26857,35
Внедрение	23,08	26857,35

Таблица 4.7 - Продолжение

Итого	**100**	**116366,31**

Так как на разработку проекта отводится 5 месяцев, то необходимо расходовать каждый месяц не более, чем 23273,26 р.

Итоговая сумма затрат, распределенная по этапам, представлена в таблице 4.8.

Таблица 4.8 – План инвестиций

Этапы реализации проекта	Инвестиции, руб.				
	I	II	III	IV	V
Техническое задание	17896,57				
Эскизный проект	5375,95	21480,54			
Технический		1791,98	16104,59		
Рабочий проект			10752,9	16104,59	
Внедрение				10751,9	16104,59
Итого	**23273,26**	**46546,52**	**69819,79**	**93093,05**	**116366,31**

Источниками финансирования являются собственные средства разработчика.

4.3. Общая характеристика организации решения задачи на ЭВМ

При внедрении разрабатываемого ПО, в работе структурных подразделений будут внесены следующие изменения:

- все сотрудники, задействованные в решении задач многокритериального выбора, будут производить вычисления с помощью онлайн-калькулятора.
- у отдела администрирования сервера глобальных изменений не произойдет.

Данные в систему будут поступать постоянно. Система рассчитана на непрерывную высоко нагруженную работу с пользователями, как на поступление данных так и на их отдачу.

4.4. Единовременные расходы организации заказчика ПО

К единовременным затратам пользователя программного обеспечения $К_{общ}$ относятся затраты на оплату:

а) программного обеспечения $Ц_{по}$;

б) инструментальных средств $Ц_{ис}$;

в) ЭВМ, прочих аппаратных средств и сетевого оборудования $К_{эвм}$;

г) обучение персонала $К_{осв}$;

д) подключения к Интернет $К_{инт}$;

г) мероприятий по организации информационной безопасности $К_{без}$.

Стоимость заказного программного обеспечения $Ц_{по}$, специально разработанного для заказчика, может быть рассчитана по формуле

$$Ц_{по} = С_{по} + П + НДС, \quad (4.7)$$

где $С_{по}$ — себестоимость ПО, затраты на разработку по смете из таблицы 4.6;

П — прибыль разработчика 30% к затратам;

НДС — налог на добавленную стоимость 18 %:

$$НДС = (С_{по} + П) \times 0,18 \quad (4.8)$$

$$НДС = (116636,31 + 34909,89) \times 0,18 = 27229,72 \text{ руб.}$$

$$Ц_{по} = 116636,6 + 34909,89 + 27229,72 = 178776,21 \text{ руб.}$$

4.4.1. Стоимость прочего программного обеспечения $К_{ис}$.

Выбор операционной системы для данного проекта основывался на следующих критериях:

1. Расширяемость.
2. Производительность.

3. Отказоустойчивость.
4. Переносимость.
5. Обратная совместимость.
6. Безопасность.

На основании этих требований была выбрана операционная система Microsoft Windows 10. Преимуществами данной операционной системы являются

- простота;
- мощность базового набора средств;
- развитость интерфейсов;
- демократичность.

Для браузера выдвинуты следующие требования:

- безопасность и секретнос;ь.
- высокая производительность. Минимальные затраты;
- соблюдение стандартов.

Исходя из этого был выбран браузер Google Chrome

Стоимость прочего программного обеспечения и инструментальных средств приведена в таблице 4.9.

Таблица 4.9 – Стоимость прочего программного обеспечения и инструментальных средств ($Ц_{ис}$)

Виды ПО	Стоимость, руб.
Microsoft Windows 10	8699,00
Google Chrome	0
Итого	**8699,00**

4.4.2. Затраты организации на освоение ПО и обучение персонала $К_{осв}$

Затраты организации на освоение ПО и обучение работе с программой и ЭВМ $К_{осв}$ можно рассчитать, упрощенно, по формуле:

$$К_{осв} = Ч_{пп} \times С_{осв} \times t_{осв}, \quad (4.10)$$

где $Ч_{пп}$ — численность персонала на обучение, чел.;

$С_{осв}$ — стоимость обучения работе на ПК в единицу времени по коммерческим расценкам (руб./нед., руб./день, руб./мес.);

$t_{осв}$ — время обучения;

$$К_{осв} = 1000 \times 100 \text{ руб./час.} \times 1 \text{ час.} = 100 \text{ т. руб.}$$

Стоимость подключения к Интернет $К_{инт}$

$$К_{инт} = 10000 \text{ руб.}$$

4.4.3. Затраты на внедрение мероприятий по информационной безопасности

Затраты на внедрение мероприятий по информационной безопасности $К_{без}$ уже включены в проект и дополнительных затрат не требуют:

$$К_{общ} = Ц_{по} + К_{инт} + Ц_{ис} + К_{осв} + К_{без} \quad (4.11)$$

$$К_{общ} = 178776{,}21 + 10000 + 8699 + 100000 + 0 = 297475{,}21 \text{ руб.}$$

Распределение инвестиций по времени реализации проекта осуществляется на основе предварительных расчетов времени, необходимого для разработки ПО по отдельным стадиям проектирования, затрат на разработку и общей суммы единовременных капитальных вложений.

Результаты расчетов оформить в виде инвестиционного плана (таблица 4.12). Затраты на разработку ПО $Ц_{по}$ распределены по периодам пропорционально трудоемкости разработки.

Таблица 4.12 – Инвестиционный план

Этапы реализации проекта инвестиций	Период				
	1	2	3	4	5
Техническое задание	27431.2				
Эскизный проект	8240.06	32924.57			
Технический проект		2746.68	24684.51		
Рабочий проект			16481.64	24684.51	
Внедрение				16480.11	24684.51
Затраты на ИС				10000	
Затраты на обучение персонала				17000,1	82999,9
Итого	**35671.26**	**71344.51**	**112510.66**	**189780.8**	**297475.21**

4.5. Текущие расходы пользователя ПО

Текущие расходы пользователя при внедрении учитывают затраты в год на:

- амортизацию оборудования, ПО и инструментальных средств;
- материалы (дискеты, картриджи и т.д.);
- эксплуатацию оборудования;
- электроэнергию;
- обтирочные и смазочные материалы;
- заработную плату техников, с отчислениями;
- ремонт оборудования (5% от стоимости);

- заработную плату операторов, системных программистов, с отчислениями;
- абонентскую плату при использовании услуг Интернет;
- сопровождение программного обеспечения.

Расчѐт амортизационных отчислений приведен в таблице 4.13.

Таблица 4.13 – Расчет амортизационных отчислений

Наименование оборудования и ПО	Стоимость, тыс. руб.	Норма амортизации, %	Амортизационные отчисления, тыс. руб.
Сервер IBM Express×3300 M4 Tower	12218,1	20%	2443.22
Итого			**$А_{эвм}$ =2443.22**
Заказное ПО Прочее ПО	178776,21	5%	8938,81
Итого			**$А_{по}$ = 8938,81**

$$Ц_{тек} = Ц_{эвм} + Ц_{по} + Ц_{рем} + Ц_{инт} + Ц_{зп} + Ц_{эл} \quad (4.12)$$

$Ц_{тек}$ = 2443,22 + 14986,06 + 14986,06 + 10000 + 7200 + 24×0,5×3,53×365 = 53161,48 руб. в год

4.6. Экономия текущих затрат пользователя ПО

Изменение годовых эксплуатационных расходов зависит от вида и назначения программного средства. Рассмотрим возможные направления экономии текущих затрат применения ПС для:

- автоматизации программирования (системное программирование);
- автоматизации рабочих мест специалистов (АРМ);

- внедрения АСУ;
- внедрения Интернет технологий.

4.6.1. Автоматизация процесса программирования

Основными источниками экономии в этом случае являются:

- сокращение текущих затрат за счет снижения расхода машинного времени, необходимого на отладку программ и сдачу их в эксплуатацию;
- экономия и улучшение использования машинных ресурсов (памяти, увеличение скорости обработки информации);
- сокращение сроков разработки программ.

4.6.2. Автоматизация рабочих мест (АРМ) специалистов и внедрение АСУ

Может обеспечить экономию за счет:

- сокращения трудоемкости выполнения работ;
- сокращения трудоемкости обработки информации;
- увеличения оперативности управления, скорости принятия решений;
- совершенствования процессов сбора, передачи, обработки, хранения, защиты целостности и секретности информации;

Прямой эффект от внедрения АРМ специалистов характеризуется, в первую очередь, снижением трудоемкости выполнения основных технологических операций на рабочем месте. Анализ трудоемкости выполнения работ при базовом варианте и в случае внедрения АРМ можно представить в форме таблицы 4.14.

Таблица 4.14 – Анализ трудоемкости выполнения работ

Наимено-вания задачи, операции	Переодич-ность выполнения	Трудоемкость работы, чел.-ч				Отклоне-ние в год,
		база		проект		
		на одну операцию	в год	на одну операцию	в год	чел.-ч, +/–
Внесение данных	200 в день	0,015788904	1152,59	0,001388889	101,39	22,3

Таблица 4.14 - Продолжение

Вычисления	200 в день	0,015788904	1152,59	0,001388889	101,39	32,2
Итого $T_{база}$ = 2305,18				$T_{пр}$ = 202,78		ΔT = 2102,4

В результате сокращения трудозатрат можно рассчитать экономию $Э_{зп}$:

$$Э_{зп}=Т\times З\times(1+Х100)\times(1+У100) \quad (4.13)$$

Итого экономия составит:

$$Э_{зп} =2102,4 \times 100\times 1,1\times 1,3=300643,2 \text{ руб.}$$

4.6.3. Внедрение Интернет-технологий

Внедрение Интернет-технологий позволяет организациям увеличивать объемы предоставляемых услуг

- за счет электронной коммерции;
- улучшения имиджа фирмы;
- получения максимального доступа к внешней информации;
- сокращения времени обработки и получения оперативных данных для принятия управленческих решений и т.д.

4.7. Финансовый план проекта

Оценка финансовой состоятельности проекта представлена в таблице 4.15. Международная практика в процессе оценки проектов использует несколько обобщающих показателей. К таким показателям относятся:

- интегральный экономический эффект (чистая текущая стоимость);
- индекс доходности;
- внутренний коэффициент эффективности;
- максимальный денежный отток;
- период возврата капитальных вложений и срок окупаемости.

4.7.1. Интегральный экономический эффект (NPV – Net Present Value of Discounted Cash Flow)

Чистая приведённая стоимость (чистая текущая стоимость, чистый дисконтированный доход, англ. Net present value, принятое в международной

практике для анализа инвестиционных проектов сокращение – NPV или ЧДД) – это сумма дисконтированных значений потока платежей, приведённых к сегодняшнему дню.

Таблица 4.15 – Оценка финансовой состоятельности проекта

Показатели, т.р.	Период															
	1..10	11	12	13	14	15	16	17	18	19	20	21	22	2-й год	3-йгод	
I.Инвестиционная деятельность																
Затраты на разработку																
Техническое задание, т.р.	17896,57															
Эскизный проект, т.р.	26857,35															
Технический проект, т.р.	17897,13															
Рабочий проект, т.р.	26857,35															
Внедрение, т.р.	26857,35															
Итого	116366,3															
II. Операционная деятельность:																
Выручка от реализации, т.р.	0	5	5	10	10	15	15	25	30	35	45	60	65	300	215	
Затраты на тиражирование, т.р.	0	1,25	1,25	2,5	2,5	3,75	3,75	6,25	7,5	8,75	11,25	15	16,25	75	53,75	
Валовая прибыль, т.р.	0	11,25	11,25	22,5	22,5	33,75	33,75	56,25	67,5	78,75	101,25	135	146,25	675	483,75	
Налог на прибыль, т.р.	0	2,25	2,25	4,5	4,5	6,75	6,75	11,25	13,5	15,75	20,25	27	29,25	135	96,75	
Прибыль чистая, т.р.	0	9	9	18	18	27	27	45	54	63	81	108	117	540	387	
III. Финансовая деятельность:																
Кредит, т.р.	123,9															
Возврат кредита, т.р.	0	0	10,3	10,3	10,3	10,3	10,3	10,3	10,3	10,3	10,3	10,3	10,3			
Итого: эффект от финансовой деятельности	0	0	-10,3	-10,3	-10,3	-10,3	-10,3	-10,3	-10,3	-10,3	-10,3	-10,3	-10,3			
IV. Сальдо денежной наличности, т.р.																
	0,1	9	-1,3	7,7	7,7	16,7	16,7	34,7	43,7	52,7	70,7	97,7	106,7	540	387	
V. Сальдо денежной наличности нарастающим итогом, т.р.																
	0,1	9,1	7,8	15,5	23,2	39,9	56,6	91,3	135	187,7	258,4	356,1	462,8	1002,8	1389,8	

Показатель NPV представляет собой разницу между всеми денежными притоками и оттоками, приведёнными к текущему моменту времени (моменту оценки инвестиционного проекта). Он показывает величину денежных средств, которую инвестор ожидает получить от проекта, после того, как денежные притоки окупят его первоначальные инвестиционные затраты и периодические денежные оттоки, связанные с осуществлением проекта. Поскольку денежные платежи оцениваются с учётом их временно́й стоимости и рисков, NPV можно интерпретировать как стоимость, добавляемую проектом. Её также можно интерпретировать как общую прибыль инвестора. Коэффициент дисконтирования определяется по формуле:

$$t = 1 \times (1+r) \times t, \quad (4.14)$$

где r – ставка дисконтирования.

Ставка дисконтирования годовая может быть рассчитана по формуле:

$$R_{год} = \frac{\text{ставка рефинансирования} + 1}{\text{инфляция в год} + 1} - 1 + \text{риск} = \frac{1.0825}{1.11} - 1 + 0.15 \approx 0.12 \quad (4.15)$$

Если необходимо осуществить дисконтирование по полугодиям, кварталам или месяцам, то ставка дисконтирования определяется по формуле:

$$R_t = r_{год} 12 = 0.1 \quad (4.16)$$

Расчет интегрального экономического эффекта NPV можно представить в виде формулы:

$$NPV = (Д_t - З_t + A_t) * \alpha_t - \sum_{t=1}^{n} K_t * \alpha_t = \sum_{t=1}^{n} (П_{ч_t} + A_t) * \alpha_t - \sum_{t=1}^{n} K_t * \alpha_t \quad (4.17)$$

где $Д_t$ – доходы от реализации проекта по годам;

$З_t$ – текущие затраты без амортизации;

A_t – амортизационные отчисления;

K_t – капитальные вложения в основные и оборотные фонды;

ПЧ – прибыль чистая.

Результаты расчета NPV представлены в таблице4.16.

Таблица 4.16 – Денежные потоки для оценки эффективности проекта, руб.

	Период														
Потоки	1-10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Инвестиции, т.р.	-116,4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Прибыль, т.р.	0	11,25	11,25	22,5	22,5	33,75	33,75	56,25	67,5	78,75	101,25	135	146,25	675	483,75
Чистый денежный поток, т.р.	-116,400	11,250	11,250	22,500	22,500	33,750	33,750	56,250	67,500	78,750	101,250	135,000	146,250	675,000	483,750
Коэффициент дисконитрования	0,936	0,875	0,865	0,854	0,844	0,834	0,824	0,814	0,804	0,794	0,785	0,775	0,766	0,757	0,748
Дисконтированный денежный поток	-108,959	9,846	9,727	19,220	18,988	28,139	27,800	45,775	54,268	62,550	79,453	104,661	112,017	510,771	361,642
NVP	-108,959	-99,113	-89,386	-70,166	-51,178	-23,039	4,761	50,536	104,805	167,355	246,808	351,469	463,486	974,257	1335,899
	R	80%	ггод	0,8	гмес	0,0666667									
Коэффициент дисконитрования	0,713	0,492	0,461	0,432	0,405	0,380	0,356	0,334	0,313	0,293	0,275	0,258	0,242	0,227	0,212
Дисконтированный денежный поток	-83,029	5,531	5,186	9,723	9,115	12,819	12,018	18,777	21,125	23,105	27,850	34,812	35,356	152,983	102,785
NVP2	-83,029	-77,498	-72,312	-62,589	-53,473	-40,655	-28,637	-9,860	11,265	34,370	62,219	97,031	132,387	285,370	388,156

4.8 Показатели эффективности проекта

Международная практика в процессе оценки проектов использует несколько обобщающих показателей. К таким показателям относят:

- интегральный экономический эффект (чистая текущая стоимость);
- индекс доходности;
- внутренний коэффициент эффективности;
- максимальный денежный отток;
- период возврата капитальных вложений и срок окупаемости.

Индекс доходности SRR — Simple Rate of Return) определяется как отношение суммарного дисконтированного дохода к суммарным дисконтированным капитальным вложениям:

$$SRR = \frac{\sum_{t=1}^{n}(П_{чr}+A_t)*\alpha_t}{\sum_{t=1}^{n}K_t*\alpha_t} = 6{,}47 \quad (4.18)$$

Внутренний коэффициент эффективности проекта — внутренняя норма доходности — (IRR — Internal Rate of Return) определяется как пороговое значение рентабельности, при котором NPV равно нулю.

$$r_{пор} = r_1 + \left[\frac{NPV_{r1}}{(NPV_{r1}-NPV_{r2})}\right] \times (r_2 - r_1) \quad (4.19)$$

$$r_{пор} = 0{,}12 + \left[\frac{1360{,}891}{(1360{,}891-388{,}156)}\right] \times (0{,}6 - 0{,}12) = 0{,}79 \quad (4.20)$$

где r_1 — исходная ставка дисконтирования; r_2 — ставка дисконтирования, при которой NPV < 0; $r_{пор}$ — внутренний коэффициент эффективности проекта; NPV_{r1}, NPV_{r2} — NPV соответственно при r_1 и r_2.

Проект считается эффективным, если $r_{пор} > r_1$.

В данном случае $r_{пор} = 0{,}79$ больше, чем $r_1 = 0{,}12$, следовательно, проект является эффективным.

Срок возврата капитальных вложений и срок окупаемости могут быть определены аналитическим и графическим способами:

а) аналитический способ:

$$T_{ок} = t_x + \frac{|NPV_t|}{ДДП_{t+1}} \quad (4.21)$$

$$T_{ок} = 15 + \frac{22{,}571}{28{,}162} = 15{,}80 \quad (4.22)$$

б) графический способ на основе построения финансового профиля проекта. Финансовый профиль проекта представляет собой график изображения величины кумулятивной чистой текущей стоимости во времени (рисунок 4.1).

Рисунок 4.1 — Финансовый профиль проекта

На основании графика финансового профиля можно определить срок возврата и срок окупаемости капитальных вложений проекта, максимальный денежный отток и интегральный экономический эффект.

Необходимыми условиями принятия инвестиционного проекта являются:

- положительное сальдо реальных накопленных денег в любом временном интервале, где участник осуществляет затраты или получает доходы;
- положительность интегрального экономического эффекта (NPV > 0);
- индекс доходности больше 1 (SRR > 1).

ЗАКЛЮЧЕНИЕ

В данной ВКР разработан онлайн калькулятор для определения рейтингов сложных объектов по комплексу показателей.

В результате выполнения работы были рассмотрены существующие методы принятия решений.

Был разработан онлайн калькулятор для ранжирования объектов, который соответствует поставленному техническому заданию.

В третьей главе работы произведено тестирование функциональности разработанной системы, результаты которого помогли обнаружить и устранить ошибки, допущенные на этапе проектирования системы.

Произведен расчет экономических показателей проекта.

Поставленные цели и задачи выпускной квалификационной работы были выполнены.

Список литературы

1. Костров, А.В. Информационный менеджмент/А.В. Костров.- М.: Финансы и статистика, 2005.
2. Гринберг, А.С. Информационный менеджмент/А.С. Гринберг.- М.: ЮНИТИ, 2005.
3. Костров, А.В. Уроки информационного менеджмента/А.В. Костров.- М.: Финансы и статистика. М.: Финансы и статистика, 2005.
4. Скопин, И.Н. Основы менеджмента программных проектов/ И.Н. Скопин.- М.: Интернет – университет информационных технологий, 2004.
5. Грабауров, В.А. Информационные технологии для менеджеров/ В.А. Грабауров.- М.: Финансы и статистика, 2005.
6. Бурдаков В.П. Термодинамика. Уч. пособие. В 2-х частях. 2009 год.
7. А.И. Орлов Теория принятия решений. 2006 год.
8. В.Г. Халин Теория принятия решений.
9. Дженнифер Нидерст Роббинс «HTML5, CSS3 и JavaScript. Исчерпывающее руководство». 4-ое издание. 2014 год.
10. Брайан Хоган «HTML5 и CSS3. Веб-разработка по стандартам нового поколения». 2011 год.
11. Дэвид Флэнаган . JavaScript. Подробное руководство. 2013 год.
12. Стоян Стефанов. JavaScript. Шаблоны. 2011 год.
13. Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. Алгоритмы. Построение и анализ. 2012 год.
14. С. Макконнел. Совершенный код. 2010 год.
15. Р. Мартин. Чистый код.

Приложение

```
document.getElementById('c1').checked=true;
var h=0;
var array;
var result_array;
var goroda;
var s;
var gorgorod;
var m_array;
var r_array;
var g;
var p = [0, 0, 0];
// Проверяем, поддерживает ли браузер нужные API
if (window.File && window.FileReader && window.FileList && window.Blob) {
   document.getElementById('file').addEventListener('change', function(e) {
    // Если бы у input[type=file] был параметр multiple, пользователь смог бы выбрать
    // несколько файлов, а в e.target.files было бы больше одного элемента
    var file = e.target.files[0];
    // Для демонстрации прочитаем файл и выведем каждую его строку как элемент списка
    var output = document.getElementById('output');
```

```
var reader = new FileReader();
// Устанавливаем обработчик события onload. Оно произойдёт по окончанию чтения файла
reader.onload = function(e) {
    // e.target.result содержит всё содержимое файла
    var text = e.target.result;
    // Разбиваем строку на элементы, разделителем служит исмвол перевода строки \n
    array = text.split('\n');
    // Для демонстрации того, что файл прочитался и разбился на строки правильно,
    // рисуем список, элементами которого будут строки файла
      result_array=new Array(array.length);
      for(var i=0; i<array.length; i++)
      result_array[i]=array[i].split(';;');
goroda = new Array(array.length);
gorgorod=new Array(array.length);
m_array=new Array(4);
for (var i=0; i<4; i++)
{
   m_array[i]=new Array(array.length);
}
for (var i=0; i<array.length; i++) {
   goroda[i]=result_array[i][0];
            }
```

```
for (var i=0; i<array.length; i++) { gorgorod[i]=goroda[i]; }
        for (var i=0; i<array.length; i++) {
   m_array[0][i]=result_array[i][1].split(';');
                                                    }

   for (var i=0; i<array.length; i++) {
   m_array[1][i]=result_array[i][2].split(';');
                                                    }
for (var i=0; i<array.length; i++) {
   m_array[2][i]=result_array[i][3].split(';');
                                                    }
for (var i=0; i<array.length; i++) {
   m_array[3][i]=result_array[i][4].split(';');
                                                    }

for (var i=0; i<m_array.length; i++)
{
   for (var i1=0; i1<m_array[i].length; i1++)
   {
        for (var i2=0; i2<m_array[i][i1].length; i2++)
        {
             m_array[i][i1][i2]=parseFloat(m_array[i][i1][i2]);
        }
   }
```

```
        }
         };
        // Начинаем чтение выбранного файла
        reader.readAsText(file);
         });
} else {
   alert('File API is not supported!');
}
function test(){
          start();
                              }
 checkboxes = [ c1, c2, c3, c4 ];
 checkboxes1 = [p1, p2, p3, p4, p5, p6, p7, p8, p9];
function check(checkBox)
{    g=parseInt(checkBox.value)-1;
     for(var i=0; i<checkboxes.length; i++){
          if(checkboxes[i].name!=checkBox.name)
                    checkboxes[i].checked=false;
     }
     if (checkboxes[3].checked==true)
     {
                    document.getElementById('p8').style.visibility = "visible" ;
            document.getElementById('p9').style.visibility = "visible" ;
     }
```

```
    else {
            document.getElementById('p8').style.visibility = "hidden" ;
        document.getElementById('p9').style.visibility = "hidden" ;
    }
}
var k=0;
function check2(checkBox)
{
    if (checkboxes1[parseInt(checkBox.value)-1].checked==true)
    {
        for (var i=0; i<3; i++)
        {
            if (p[i]==0) {p[i]=parseInt(checkBox.value);}
        }
        k++;
    }
    else
    {
        for (var i=0; i<3; i++)
        {
            if (p[i]==parseInt(checkBox.value)) {p[i]=0;}
        }
        k--;
    }
```

```
}
 function start()
 {
     if (k==3)
     {
     for (var i=0; i<3; i++) {p[i]--;}
     r_array=new Array(m_array[g].length);
         for (var i=0; i<m_array[g].length; i++) {r_array[i]=new Array(3);}
         for (var i=0; i<m_array[g].length; i++)
         {
                 for (var j=0; j<3; j++)
                 {
                         r_array[i][j]=m_array[g][i][p[j]];
                 }
         }
     main();
     }
     else {alert('Выберите 3 параметра');}
 }
 function main()
 {
      s = new Array(array.length);
      var del=1;
      for (var i=0; i<3; i++)
```

```
    {
        del=del*r_array[0][i];
    }
      for (var i=0; i<array.length; i++) {
            s[i]=1;
            for (var j=0; j<3; j++) {
                  s[i]=s[i]*r_array[i][j];
            }
            s[i]=s[i]/del;
      }
   sort();
}
 function sort()
 {
     //alert(s[4]);
     var x;
     for(var i=0; i < s.length; i++) {          // i - номер прохода
  for(var j = s.length-1; j > i; j-- ) {     // внутренний цикл прохода
   if ( s[j-1] < s[j] ) {
   x=s[j-1]; s[j-1]=s[j]; s[j]=x;
     x=gorgorod[j-1]; gorgorod[j-1]=gorgorod[j]; gorgorod[j]=x;
     }}}
   vivod();
 }
```

```
  function vivod() {
var viw = document.getElementById('io');
var lo=' ';
viw.innerHTML = '<ul>';
for (var i=0; i<array.length; i++)
{
     viw.innerHTML += '<li>' + gorgorod[i] + ' ' + s[i] + '</li>';
}
viw.innerHTML += '</ul>';
}
```